



日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日

Date of Application:

2001年 4月24日

出願番号

Application Number:

特願2001-125224

[ST.10/C]:

[JP2001-125224]

出願人

Applicant(s):

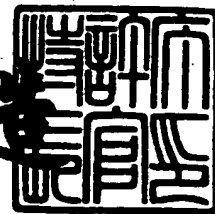
株式会社日立製作所

CERTIFIED COPY OF
PRIORITY DOCUMENT

2002年 2月15日

特許庁長官
Commissioner,
Japan Patent Office

及川耕造



出証番号 出証特2002-3007589

【書類名】	特許願
【整理番号】	K01007871A
【あて先】	特許庁長官
【国際特許分類】	G06F 9/46
【発明者】	
【住所又は居所】	神奈川県川崎市麻生区王禅寺 1 0 9 9 番地 株式会社日立製作所 システム開発研究所内
【氏名】	宮崎 邦彦
【発明者】	
【住所又は居所】	神奈川県川崎市麻生区王禅寺 1 0 9 9 番地 株式会社日立製作所 システム開発研究所内
【氏名】	伊藤 信治
【発明者】	
【住所又は居所】	神奈川県川崎市麻生区王禅寺 1 0 9 9 番地 株式会社日立製作所 システム開発研究所内
【氏名】	吉浦 裕
【発明者】	
【住所又は居所】	神奈川県川崎市麻生区王禅寺 1 0 9 9 番地 株式会社日立製作所 システム開発研究所内
【氏名】	宝木 和夫
【発明者】	
【住所又は居所】	神奈川県川崎市麻生区王禅寺 1 0 9 9 番地 株式会社日立製作所 システム開発研究所内
【氏名】	荒井 正人
【発明者】	
【住所又は居所】	神奈川県川崎市麻生区王禅寺 1 0 9 9 番地 株式会社日立製作所 システム開発研究所内
【氏名】	新井 利明

【発明者】

【住所又は居所】 神奈川県川崎市幸区鹿島田 8 9 0 番地 株式会社日立製作所 情報サービス事業部内

【氏名】 松木 武

【発明者】

【住所又は居所】 東京都江東区新砂一丁目 6 番 2 7 号 株式会社日立製作所 公共システム事業部内

【氏名】 豊島 久

【特許出願人】

【識別番号】 000005108

【氏名又は名称】 株式会社日立製作所

【代理人】

【識別番号】 100075096

【弁理士】

【氏名又は名称】 作田 康夫

【手数料の表示】

【予納台帳番号】 013088

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 高信頼計算機システム

【特許請求の範囲】

【請求項1】 ファイルアクセスを実行する入出力処理部と、

ユーザの指示により、前記入出力処理部へファイルアクセスを要求するアクセス実行処理部と、

前記ファイルアクセス実行時にアクセス制御を行うアクセス制御部を備える計算機システムであって、

前記アクセス制御部は、

前記アクセス実行処理部から保護された記憶部と、

前記ファイルのセキュリティレベルを記述した、前記記憶部中のファイルリストと、

前記ユーザのクリアランスを記述した、前記記憶部中のユーザリストと、

前記ファイルアクセスの正当性を、前記ファイルリストと、前記ユーザリストと、前記ファイルアクセスのアクセスタイプと、前記ファイルを特定する情報と、前記ユーザを特定する情報と、に基づき判定するアクセス制御処理部と、

前記入出力処理部によるファイルアクセスの実行時に、前記アクセスタイプと、前記ファイルを特定する情報と、前記ユーザを特定する情報と、を前記アクセス制御処理部に送り、

前記ファイルアクセスの正当性判定結果を前記アクセス制御処理部から受け取り、

前記ファイルアクセスが正当であれば、前記入出力処理部によるファイルアクセスを実行させ、不当であれば前記ファイルアクセスを中止させるアクセス監視処理部と、を備える

アクセス制御部を備える計算機システム。

【請求項2】 請求項1に記載のアクセス制御部を備える計算機システムであって、

前記アクセス制御処理部が使用する前記記憶部中の記憶領域を、前記アクセス実行処理部から保護するための排他制御部を備える

アクセス制御部を備える計算機システム。

【請求項3】請求項2記載のアクセス制御部を備える計算機システムであって、
前記ユーザリストの設定と管理を行うためのユーザリスト設定管理処理部を備える

アクセス制御部を備える計算機システム。

【請求項4】請求項3記載のアクセス制御部を備える計算機システムであって、
前記ユーザリスト設定処理部は、セキュリティ管理者の認証処理部を備える
アクセス制御部を備える計算機システム。

【請求項5】請求項4記載のアクセス制御部を備える計算機システムであって、
前記セキュリティ管理者は、前記アクセス実行処理部を管理するシステム管理者とは異なる

アクセス制御部を備える計算機システム。

【請求項6】請求項1記載のアクセス制御部を備える計算機システムであって、
前記ファイルリストの設定と管理を行うためのファイルリスト設定管理処理部を備える

アクセス制御部を備える計算機システム。

【請求項7】請求項6記載のアクセス制御部を備える計算機システムであって、
前記ファイルリスト設定管理処理部は、セキュリティ管理者の認証処理部を備える

アクセス制御部を備える計算機システム。

【請求項8】請求項7記載のアクセス制御部を備える計算機システムであって、
前記セキュリティ管理者は、前記アクセス実行処理部を管理するシステム管理者とは異なることを特徴とする、

アクセス制御部を備える計算機システム。

【請求項9】請求項1記載のアクセス制御部を備える計算機システムであって、
記憶媒体へファイルの出力を要求するファイルアクセスが正当であれば、前記
ファイルを暗号化する暗号処理部と、

記憶媒体から暗号化されたファイルの入力を要求するファイルアクセスが正当
であれば、前記暗号化されたファイルを復号化する復号処理部を備える

アクセス制御部を備える計算機システム。

【請求項 1 0】請求項 9 記載のアクセス制御部を備える計算機システムであって

前記排他制御部は、前記暗号処理部と復号処理部で利用される、少なくともひとつの鍵情報が記憶される前記記憶部中の記憶領域を前記アクセス実行処理部から保護する

アクセス制御部を備える計算機システム。

【請求項 1 1】請求項 9 記載のアクセス制御部を備える計算機システムであって

前記暗号処理部と復号処理部は、前記ファイルリストに記述されたセキュリティレベル毎に異なる、複数の鍵情報と少なくとも一つの暗号方法を利用するアクセス制御部を備える計算機システム。

【請求項 1 2】請求項 1 記載のアクセス制御部を備える計算機システムであって

前記入出力処理部またはアクセス監視処理部が、改ざんされていないこと、あるいはあらかじめ定められた動作をしていることを監査し、前記入出力処理部またはアクセス監視処理部が改ざんされている場合、あるいはあらかじめ定められた動作と異なる動作をしている場合には、ファイルの入出力を中止を指示する入出力処理監査処理部を備える

アクセス制御部を備える計算機システム。

【請求項 1 3】請求項 1 記載のアクセス制御部を備える計算機システムであって

前記アクセス制御処理部に伝えられたファイルアクセスに関する情報を記録し、管理するファイルアクセスログ処理部を備える

アクセス制御部を備える計算機システム。

【請求項 1 4】請求項 1 記載のアクセス制御部を備える計算機システムであって

前記アクセス制御部は、ソフトウェアモジュールとして実現されているアクセス制御部を備える計算機システム。

【請求項 1 5】請求項 1 記載のアクセス制御部を備える計算機システムであって

前記アクセス制御部は、ハードウェアモジュールとして実現されているアクセス制御部を備える計算機システム。

【請求項 1 6】各種処理に必要な情報を格納するための記憶部と、

暗号の復号化処理部、または、デジタル署名の生成処理部を含む、暗号機能処理部と、

前記暗号機能処理部で利用される鍵情報を保持する鍵記憶部と、

ユーザの指示により、暗号機能実行を要求する暗号利用要求実行処理部と、

前記暗号利用要求実行処理部から前記暗号機能処理部に対する暗号機能実行要求を、監視して前記暗号機能処理部に伝え、処理結果を前記暗号機能処理部から受信する暗号利用要求監視処理部と、

前記暗号機能処理部が使用する前記記憶部中の記憶領域、または、鍵記憶部を、前記暗号利用要求実行処理部から保護するための排他制御部とを備える計算機システム。

【請求項 1 7】請求項 1 6 記載の計算機システムであって、

前記暗号機能処理部は、前記ユーザの認証処理部を備える計算機システム。

【請求項 1 8】請求項 1 6 記載の計算機システムであって、

前記暗号機能処理部は、ソフトウェアモジュールとして実現されている計算機システム。

【請求項 1 9】請求項 1 6 記載の計算機システムであって、

前記暗号機能処理部は、ハードウェアモジュールとして実現されている計算機システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は情報処理システムに保管されているファイルへの不正アクセスを防止し、高い安全性を確保するための技術に関する。

【0002】

【従来の技術】 計算機を用いた情報処理システムにおいては、多くのアプリケーションプログラムを実行させるための基本的な仕組みを提供するOS (Operating System)を用いてハードウェア資源の有効活用を図るのが一般的である。

【0003】

さらに、当該情報処理システムでのファイルの保護は、主として、上記OSが備える一機能であるファイルアクセス制御機能を用いて実現している。

現在、広範に利用されているOSでは、ファイルアクセス制御を、任意アクセス制御(Discretionary Access Control、DACとも記す)に基づいて行うことが多い。

これは、ファイルの所有者(一般には、ファイルの作成者と一致する)であるユーザが、当該ファイルに対して、アクセスできるユーザ、および、アクセスの種類(読み取りのみ、読み取りと変更など)を任意に設定し、それに基づいてアクセス制御を行うものである。

また、通常このようなOSでは、システム管理者と呼ばれる特殊な権限をもつユーザは、当該情報処理システム内のすべての設定を変更することが可能である。

【0004】

一方、高いセキュリティが要求される場面での利用を想定して作られた専用OSの中には、より厳格なアクセス制御方法である、強制アクセス制御(必須アクセス制御とも言う)(Mandatory Access Control、MACとも記す)に基づくファイルアクセス制御機能を備えたものがある。たとえば、セキュアシステムの評価基準のひとつである米国のTCSEC(Trusted Computer System Evaluation Criteria)に定められたクラスのうち、クラスB1以上に認定されるためには、強制アクセス制御機能を備えなければならない。

TCSECおよび強制アクセス制御については、
文献[TCSEC]:「Department of Defense Trusted Computer System Evaluation Criteria」アメリカ国防総省標準(DOD 5200.28-STD)
に詳しい。

【0005】

強制アクセス制御を備えた情報処理システムにおいては、すべてのファイルには、セキュリティレベル(例えば、高い方から「機密(top secret)」「極秘(secre

t)」「秘密 (confidential)」「一般 (unclassified)」など)が設定され、また、すべてのユーザに対し、信頼度、すなわち、アクセスできるファイルのセキュリティレベルを示す許可レベル(クリアランス (clearance) という。例えば、高い方から「機密」「極秘」「秘密」「一般」など))が設定される。システムは、これらの情報に基づいて強制的にアクセス制御を行う。

【0006】

上記のような専用OSでは、システム管理者とは異なる、セキュリティ管理者とよばれるユーザによって、ファイルのセキュリティレベルや他のユーザのクリアランスなど、セキュリティに関わる機能が管理されていることが多い。これにより、ファイル作成者だけでなくシステム管理者であっても当該情報処理システムのセキュリティを下げることはできないようになり、より安全なシステムの運用が可能となる。

【0007】

【発明が解決しようとする課題】

ファイルへの不正アクセスを防ぐ仕組みを任意アクセス制御に基づくアクセス制御機能を備えたOSに依存している情報処理システムでは、各ファイルの所有者であるユーザが、当該ファイルのアクセス可能な範囲を設定できてしまうため、ユーザの故意、または、不注意により、重要な情報が漏洩してしまう恐れがある。

【0008】

一方、強制アクセス制御機能を備えたOSを利用した情報処理システムでは、当該OSが一般的ではないため、ユーザが利用するアプリケーションプログラムを独自に開発する必要があり、コストがかさむ上、ユーザの利便性が失われる。

上記背景の元、コストがかからず、安全性を高められる技術が求められている。

【0009】

【課題を解決するための手段】

本発明は、情報処理システムの情報セキュリティに関する安全性を、利用者の利便性を損なうことなく、向上させる技術を提供する。

本発明は、また、情報処理システム内にシステム管理者とは異なる、資格を持つユーザだけが利用できる機能・領域を設けるための方法、および、装置を提供する。

本発明は、また、上記技術を適用した情報処理システムに対する、起動時における不正を防止するための方法、および、装置を提供する。

【 0 0 1 0 】

より具体的には、既存のアプリケーションプログラムを利用可能な状態を保ちつつ、情報処理システムに強制アクセス制御機能を付加する技術と、その技術を適用した情報処理システムを提供する。

また、強制アクセス制御機能をもたないOSの管理の元で既存のアプリケーションプログラムが動作する情報処理システムに、強制アクセス制御機能を付加する技術と、その技術を適用した情報処理システムを提供する。

【 0 0 1 1 】

本発明の一態様によれば、ユーザが直接利用する、アプリケーションプログラムを動作させるためのOS(ホストOSという)と、セキュリティ強化のためのOS(ゲストOSという)の2つのOSを、複数OS制御技術を利用することにより動作させる。複数OS制御技術は、たとえば、特開平11-149385号公報に開示されている。

本発明に従えば、上記専用OSを用いることなく、ホストOS上で動作するファイル作成者だけでなく、システム管理者であっても、セキュリティレベルを変えることが非常に困難になる。

【 0 0 1 2 】

より具体的には、アクセス監視処理プログラムはホストOS管理下で行われるファイルアクセスを監視し、ゲストOS上のファイルアクセス制御プログラムは、アクセス監視処理プログラムが検知したファイルアクセスの正当性を、ファイルのセキュリティレベルと、ユーザのクリアランスに従って判断し、不正アクセスを許さない。

また、ホストOSの各種設定を行うシステム管理者とは、別に、セキュリティ管理者を設け、システム管理者であっても、セキュリティレベルを変更できないようにする。システム全体のセキュリティポリシーとして、システム管理者が、セ

セキュリティ管理者の権限を有することが許される場合には、これら2つの管理者を同一人が兼任してもよい。

【0013】

本発明の他の一態様によれば、ゲストOS上に、アクセス制御機能を設けることにより、アクセス制御機能に対する不正操作を防止する。

また、本発明の他の一態様によれば、ホストOS側の多機能化・高機能化等の改良が行われても、強制アクセス制御が可能な環境を維持することを可能とする。

さらに、ユーザのクリアランスやファイルのセキュリティレベルの設定にあたっては、セキュリティ管理者の認証を行うことにより、ホストOS側の管理者による不正の脅威を減らすことができるようになる。

【0014】

なお、本発明におけるOSとは、ユーザーまたはプログラムからの要求に応じて記憶媒体中のデータ、ファイルへのアクセスを実行する機能と、アクセス要求元のユーザーやプログラムを識別する機能と、排他制御機能により占有する記憶部を有するプログラムモジュールを意味しており、

(1) ファイルアクセスを管理しており、検知が可能である。

(2) アクセス要求元のユーザーを識別できる。

(3) アクセス要求元のアプリケーションプログラムを識別できる。

といった特徴を持つ。したがって、上記特徴を持つものであれば、本発明のOSに含めることが可能である。

【0015】

【発明の実施の形態】図1は、本実施例による情報処理システムの構成例である。コンピュータ1001内には、コンピュータ内の各OSや各プログラム（処理部とも言う）を実行するCPU1003、ホストOS1017が管理するメモリA1005、および、ゲストOS1019が管理するメモリB1007があり、さらに、キーボード1011、マウスなどの入力装置、ディスプレイ1009などの出力装置、ホストOS1017の管理下にあるハードディスクA1013、ゲストOS1019の管理下にあるハードディスクB1015などの記憶装置がつながっている。なお、記憶装置は、ハードディスクの他、フラッシュメモリや、EEPROMなど書

き換え可能な不揮発性メモリであってもよい。さらには、ハードウェア的な耐タンパ性を持っているほうが望ましい。

【0016】

メモリA1005内には、ホストOS1017と、さらにホストOS1017管理下で動作する一般のアプリケーションプログラム1021の他、本実施例では、ユーザのクリアランスを定義するラベル付けを行うユーザラベル設定プログラム1023、各ファイルのセキュリティレベルを定義するラベル付けを行うファイルラベル設定プログラム1025が、ハードディスクA1013からロードされる。また、ホストOS1017内には、ファイル管理を行う部分として、I/Oマネージャ1027、ファイルシステムドライバ1031、デバイスドライバ1033がある他、本実施例では、さらに、アプリケーションからハードディスクA1013内のファイル1045へのアクセスをフックする、ファイルアクセスフックプログラム1029が設けられる。

【0017】

ファイルへのアクセスをフックするとは、ハードディスクA1013等の記憶媒体内のファイルへのアクセスが発生しようとしていることを、何らかの方法によって検知することである。

ファイルアクセスのフックは、たとえば、あらかじめOSによって用意されたインターフェイス、たとえばファイルシステムドライバ1031やデバイスドライバ1033などの各種ドライバ用に用意されたインターフェイスを利用することが可能である。

すなわち、本実施例のように、I/Oマネージャ1027と、ファイルシステムドライバ1031との間に、I/Oマネージャ1027からは、ファイルシステムドライバ1031に対して命令するよう見え、また逆に、ファイルシステムドライバ1031からは、I/Oマネージャ1027から命令されたかのように見える、ファイルアクセスフックプログラム1029を設ければ、ファイルアクセスが発生していることを、I/Oマネージャ1027による命令によって認識することができる。一般的に、OSには、各種デバイスを統一的に管理し、また、様々な機能拡張に対応することができるように、このようなインターフェイスが用意さ

れているし、さらに、ファイルアクセスフックのための専用のインターフェイスが用意されていることもある。

この他、図示されていないが、ホストOS1017は、一般にOSが備える機能を実現する部分、すなわち、プロセス管理、メモリ管理などを担う部分を備える。

【0018】

メモリB1007内には、ゲストOS1019と、さらにゲストOS1019管理下で動作する、アクセス制御プログラム1035、ユーザリスト管理プログラム1037、ファイルリスト管理プログラム1039が、ハードディスクB1015からロードされる。ゲストOS1019の管理下にあるハードディスクB1015には、ユーザリスト管理プログラム1037が利用するユーザリスト1041、ファイルリスト管理プログラム1039が利用するファイルリスト1043が含まれている。

【0019】

本実施例においては、ホストOS1017上のアプリケーションプログラム1021によるファイルアクセスは、ファイルアクセスフックプログラム1029によりフックされる。

アプリケーションプログラム1021から発行されたファイルアクセスを受けたI/Oマネージャ1027は、ファイルシステムドライバ1031に対してファイルアクセス命令を発行する。ファイルアクセスフックプログラム1029は、ファイルシステムドライバ1031の代わりにこの命令を受け取り、そのファイルアクセス命令をファイルシステムドライバ1031に渡す前に、複数OS制御プログラム1049内のOS間通信プログラム1051を用いて、ゲストOS1019上のアクセス制御プログラム1035に渡す。

【0020】

アクセス制御プログラム1035は、当該ファイルアクセスが正当なものか否かを、ユーザリスト1041、および、ファイルリスト1043を元に判定し判定結果を返す。ファイルアクセスフックプログラム1029は、ファイルアクセスが不当であれば、当該ファイルアクセスを失敗させ、正当であれば要求どおりのファイルアクセス命令をファイルシステムドライバ1031に渡す。

【 0 0 2 1 】

本実施例では、アクセス制御プログラム 1 0 3 5 を、ユーザがログインするホスト OS 1 0 1 7 管理下でなく、ゲスト OS 1 0 1 9 管理下で実行することにより、

- (1) アクセス制御プログラム 1 0 3 5 が不正な改変にさらされる脅威が少ない
- (2) アクセス制御に関するプログラムのバグ等によるセキュリティホールがあるかどうかをチェックすべき範囲を限定できるため、メンテナンスが容易になる
- (3) OS の多機能化・高機能化などによりホスト OS 1 0 1 7 が改良された場合にも、基本的な枠組みを変えずに、アクセス制御を行うことができるため、OS の開発コストの削減が可能

などの利点を得られる。

【 0 0 2 2 】

本実施例においては、ファイルのセキュリティレベルとして「一般」、「秘密」、「極秘」、「機密」の 4 段階あり、この順に機密度合いが高くなる。また、対応するユーザのクリアランスとして、同様に「一般」、「秘密」、「極秘」、「機密」の 4 段階あり、この順に信頼度が高くなる。ファイルアクセスが正当であるか否かは、ファイルのセキュリティレベルとユーザのクリアランスとに基づいて判断される。

【 0 0 2 3 】

セキュリティレベルを何段階設けるかは、システムごとに自由に設定できる。ただし、ファイルのセキュリティレベルとユーザのクリアランスとは 1 対 1 に対応するように設定する。

【 0 0 2 4 】

さらに、それぞれのファイルとユーザは「人事部門」、「営業部門」、「企画部門」、「研究部門」の 4 つの区画 (compartment) に所属させることができる。なお、区画の数はシステムごとに自由に設定できる。特に区画が 1 つの場合は、区画を設けなくてもよい。

【 0 0 2 5 】

各ユーザは、与えられたクリアランスと区画が同じで、かつ、機密度合いが同じか低いセキュリティレベルのファイルは、読み出し可能であり、与えられたクリアランスと区画が同じで、かつ、機密度合いが同じか高いセキュリティレベル

のファイルは、書き込み、または、作成可能である。

見方を変えると、各ファイルは、与えられたセキュリティレベルと機密度合いが同じか低いクリアランスのユーザによる書き込みまたは作成は可能であり、機密度合いが同じか高いクリアランスのユーザによる読み出しは可能である。

【0026】

強制アクセス制御においては、「極秘」のクリアランスを持つユーザAは、「一般」または「秘密」または「極秘」のセキュリティレベルのファイルを読むことはできるが、「機密」のセキュリティレベルのファイルを読むことはできない。また、ユーザAは、「機密」または「極秘」のセキュリティレベルのファイルに書き込むはできるが、「一般」または「秘密」のファイルに書き込むことはできない。

【0027】

このように、読み取りが許可されるファイルのセキュリティレベルと、書き込みが許可されるファイルのセキュリティレベルが異なるのは、情報(ファイル)に設定されたセキュリティレベルが下げられてしまうことを防ぐためである。仮に、ユーザAが「一般」のファイルに書き込むことができてしまうとすると、もともと「極秘」であったファイルを、「一般」のファイルとして保存することができてしまい、結果的に、本来「一般」のクリアランスしか持たないユーザに「極秘」の情報が漏洩してしまう。強制アクセス制御では、このような危険性を防止できる。

【0028】

本実施例の強制アクセス制御におけるファイルアクセスの種類(アクセスタイプという)として、ファイルを開いて内容を読み出すこと、ファイルの情報を書き換えること、新たにファイルを作成すること、を定義する。

ファイルの情報の書き換えにおいて、ファイルを開くかどうかは問わない。すなわち、「書き換えること」ができて「読み出すこと」ができないことがあり得る。つまり、読み取りできない(内容のわからない)ファイルの内容を書き換えることができる権利をもつことがありえる。これによって、クリアランスの低いユーザが高いセキュリティレベルのファイルを書き換える際に、情報が漏れることを防ぐ。

【0029】

さらに、書き換えについては、ファイルのセキュリティレベルを下げられないようにするという目的を達成するためであれば、元のファイル情報の書き換えを含む書き込みを許してもよいが、クリアランスの低いユーザによって、高いセキュリティレベルの情報を書き換えられることを避けるという観点からは、たとえば追記のみを許すようにしてもよい。

【 0 0 3 0 】

すなわち、ファイル内の情報を減らすようなファイルアクセスを「削除」と呼び、ファイル内の情報を増やすようなファイルアクセスを「追記」と呼ぶとすると、情報の不正な漏洩を防ぐという目的だけであれば、与えられたクリアランスと区画が同じで、かつ、機密度合いが同じか高いセキュリティレベルのファイルに対して、「削除」と「追記」の両方を許すようにしてもよいし、情報の改竄を防ぐという目的からは、「削除」は許さず「追記」を許すようにしてもよい(この場合、たとえば、機密度合いが同じ場合のみ「削除」を許すようにしてもよい)。

【 0 0 3 1 】

ユーザのクリアランスは、セキュリティ管理者がユーザラベル設定プログラム 1 0 2 3 を利用することによって、たとえば、ホスト OS 1 0 1 7 上で最初にユーザ登録を行うときに設定する。あるいは、当該ユーザが最初に何らかのファイルアクセスを行おうとしたときに、当該ファイルアクセスをフックすることにより、ユーザラベル設定プログラム 1 0 2 3 を自動的に起動するようにしてもよい。ユーザラベル設定プログラム 1 0 2 3 の起動が設定されていない場合には、ユーザのクリアランスが設定されていない状態で(すなわち、ゲスト OS 1 0 1 9 側で管理されているユーザリスト 1 0 4 1 に当該ユーザに関するクリアランスが登録されていない状態で)、ファイルアクセスが発生する可能性があるが、この場合には、システムのセキュリティポリシーにあわせて、たとえば、いかなるファイルアクセスも許さないようにしてもよいし、あるいは、最も低いクリアランス(本実施例の場合は「一般」)が与えられているものとみなしてもよい。

【 0 0 3 2 】

ファイルのセキュリティレベルは、ユーザが当該ファイルを作成したときに、

たとえばユーザが明示的に設定する。設定可能なセキュリティレベルは、当該ユーザのクリアランスによって決まる。すなわち、当該ユーザのクリアランスと同じかそれより高いセキュリティレベルを設定可能である。また、一度設定されたセキュリティレベルは、一般には、後から下げることにはできないようにしたほうがセキュリティの観点からは望ましい。但し、利便性向上等の目的で、特別の権限を持つユーザ、例えばセキュリティ管理者、がファイルラベル設定プログラム 1 0 2 5 を利用することにより、セキュリティレベルを下げられるようにしてもよい。

【 0 0 3 3 】

ユーザのクリアランスを設定するセキュリティ管理者と、ファイルのセキュリティレベルを設定するセキュリティ管理者は同一でもよいし、異なってもよい。さらには、それぞれの管理者は信頼できる複数の人であってもよい。たとえば、各区画ごとにユーザのクリアランスおよびファイルのセキュリティレベルを設定するセキュリティ管理者を設けてもよい。

【 0 0 3 4 】

図 2 を用いて、本実施例におけるファイルアクセス制御の処理フロー(読み出し、書き込み)を詳細に説明する。

ステップ 2 0 0 1 : はじめ

ステップ 2 0 0 2 : アプリケーションプログラム 1 0 2 1 は、I/O マネージャ 1 0 2 7 を通じ、ファイルアクセスフックプログラム 1 0 2 9 に対し、ファイルアクセスを要求

ステップ 2 0 0 3 : ファイルアクセスフックプログラム 1 0 2 9 は、OS 間通信プログラム 1 0 5 1 を利用し、アクセス制御プログラム 1 0 3 5 に、ユーザ名、ファイル名、アクセスの種類を送る。ユーザ名は、たとえば、アプリケーションプログラム 1 0 2 1 に関するプロセスとそのプロセスを実行したユーザ名との対応情報を参照することによって取得できる。この対応情報は、ホスト OS 自身の持つ機能のひとつであるプロセス管理機能を実現するために使われるもので、ホスト OS が保持している。

ステップ 2 0 0 4 : アクセス制御プログラム 1 0 3 5 は、ユーザリスト 1 0 4 1

およびファイルリスト1043を参照し、当該ユーザのクリアランス、および、当該ファイルのセキュリティレベルを調べる

ステップ2005：アクセス制御プログラム1035は、正当なアクセスか否かを判定する。正当なアクセスであれば、ステップ2008へ。不当なアクセスであれば、ステップ2006へ

ステップ2006：アクセス制御プログラム1035は「アクセス不可」を出力する。この判定結果はファイルフックプログラム1029、I/Oマネージャ1027を通じて、アプリケーションプログラム1021に伝えられる

ステップ2007：おわり（アクセス不可）

ステップ2008：アクセス制御プログラム1035は、ファイルフックプログラム1029を通じて、ファイルシステムドライバ1031に、ファイル名、アクセスの種類(アクセスタイプ)を送る

ステップ2009：ファイルシステムドライバ1031は、デバイスドライバ1033に、対象となるファイル名に応じたハードディスクA1013内のアドレス、および、アクセスの種類を送る

ステップ2010：デバイスドライバ1033は、ハードディスクA1013から読み込み対象となっているファイルを読み出してアプリケーションプログラム1021に送る（読み込み時）、あるいは、ハードディスクA1013にアプリケーションプログラム1021が書き込もうとしている情報を書き込む（書き込み時）

ステップ2011：おわり（アクセス成功）。

【0035】

図3を用いて、本実施例におけるファイルアクセス制御の処理フロー（ファイル作成）を詳細に説明する。

ステップ3001：はじめ

ステップ3002：アプリケーションプログラム1021は、I/Oマネージャ1027を通じ、ファイルアクセスフックプログラム1029に対し、ファイルアクセス（ファイル作成）を要求

ステップ3003：ファイルアクセスフックプログラム1029は、アプリケーションプログラム1021のユーザに対し、作成するファイルに設定するセキュ

リティレベルの入力を、そのためのユーザインタフェースを用いて、求める
 ステップ3004：ファイルアクセスフックプログラム1029は、OS間通信プログラム1051を利用し、アクセス制御プログラム1035に、ユーザ名、ファイル名、アクセスの種類(ファイル作成)、設定セキュリティレベルを送る。ユーザ名は、ステップ2003と同様に取得できる。

ステップ3005：アクセス制御プログラム1035は、ユーザリスト1041を参照し、当該ユーザのクリアランスを調べる

ステップ3006：アクセス制御プログラム1035は、設定しようとしているセキュリティレベルが正当か否かを判定する。正当であれば、ステップ3009へ。不当なアクセスであれば、ステップ3007へ

ステップ3007：アクセス制御プログラム1035は「アクセス不可」を出力する。この判定結果はファイルフックプログラム1029、I/Oマネージャ1027を通じて、アプリケーションプログラム1021に伝えられる

ステップ3008：おわり（アクセス不可）

ステップ3009：アクセス制御プログラム1035は、ゲストOS1019管理下のファイルリスト1043を更新し、I/Oマネージャ1027を通じて、ファイルシステムドライバ1031に、ファイル名、アクセスの種類(ファイル作成)を送る

ステップ3010：ファイルシステムドライバ1031は、ファイル名に応じたハードディスクA1013内のアドレスを割り当て、デバイスドライバ1033に、当該アドレスとアクセスの種類を送る

ステップ3011：デバイスドライバ1033は、ハードディスクA1013にアプリケーションプログラム1021が保存しようとしている情報を書き込む

ステップ3012：おわり（アクセス成功）。

【0036】

図6は、本実施例におけるユーザリスト1041の一例である。1列目にファイル名6001が示されており、2列目に当該ファイルのセキュリティレベル6003が示されている。

【0037】

図7は、本実施例におけるファイルリスト1043の一例である。1列目にユーザ名7001が示されており、2列目に当該ユーザのクリアランス7003が示されている。

【0038】

図8は、上述の強制アクセス制御の考え方に基づき、本実施例におけるユーザ”researcher”がアクセスすることが許されるファイルのセキュリティレベルを示したものである。2列目から5列目までが、区画（人事部門8001、8021、営業部門8003、8023、企画部門8005、8025、研究部門8007、8027）を、2行目から5行目までがセキュリティレベル（機密8009、8029、極秘8011、8031、秘密8013、8033、一般8015、8035）を示している。

【0039】

図7に示すように”researcher”に与えられたクリアランスは、「研究部門・極秘」および「企画部門・秘密」であるので、上述の強制アクセス制御の考え方に基づくと、読み取りについては、「研究部門・極秘、秘密、一般」「企画部門・秘密、一般」のファイルにアクセスできる。また、書き込み、または、作成については、「研究部門・機密、極秘」「企画部門・機密、極秘、秘密」のファイルにアクセスできる。したがって、たとえば、ファイル”C:\home\user2\research_report.doc”のセキュリティレベルは、「研究部門・極秘」であるので、読み書き可能であるが、ファイル”C:\home\user1\secret.txt”のセキュリティレベルは、「営業部門・極秘、企画部門・極秘」であるので、書き込みのみ可能である。

【0040】

アクセス制御プログラム1035は、図8に示す内容をルール化、あるいは、テーブル化したものをあらかじめ備え、その内容を参照しつつ、アクセス制御を行う。

図8の表では、特定のユーザがアクセス可能なファイルのセキュリティレベルを示しているが、特定のファイルへのアクセス可能なユーザを示すように作成してもよい。

【0041】

以上に示したとおり、本実施例は、ホストOS1017に、アプリケーションプログラムに優先して働く強制アクセス制御機能を付加するものである。したがって、本実施例によって、ホストOS1017上で動作するアプリケーションプログラムにもともと許されていたファイルアクセスが禁止される可能性があるが、当該アプリケーションプログラムを変更する必要はなく、そのまま使用できる。

【0042】

一例として、ホストOS1017管理下で、テキストエディタプログラムを実行させた場合の動作を述べると次のようになる。ここでは、ユーザUのクリアランスが、「研究部門」の「極秘」であるとし、ファイルF1のセキュリティレベルが「研究部門」の「機密」、ファイルF2のセキュリティレベルが「研究部門」の「秘密」であるとする。ユーザUは、テキストエディタプログラムを実行し、当該テキストエディタプログラムの「ファイルを開く」という機能を用い、ファイル名「F1」を指定し、ファイルの内容を読み込もうとした場合、これは、強制アクセス制御によれば許されないアクセスであるので、本実施例に示した機能によりこのアクセスは禁じられ、ユーザには、ファイルを開けない旨、伝えられる(表示される)。

【0043】

このユーザへの表示は、たとえば、ホストOS1017自体が備える機能を利用して行えばよい。すなわち、本実施例によるアクセス制御とは別に、ホストOS1017自体が備えるファイルアクセス制御機能において、「読み込み禁止」となっているファイルを読み出そうとしたときの表示を利用すればよい。

同様に、「ファイルを開く」機能によってファイル名「F2」を指定し、ファイルの内容を読み込もうとした場合には、許されるアクセスであるので、ファイルの内容がテキストエディタプログラムに読み込まれる(ユーザから見ると通常のOS上での動作と同じに見える)。

さらにこのファイルを編集したのち、やはりテキストエディタに備えられた「上書き保存」機能を利用して、ファイル名「F2」として上書きしようとする、これは許されないアクセスであるので、このアクセスは禁じられ、ユーザにはファイルが保存できない旨、伝えられる(表示される)。読み込みの場合と同様に、

このユーザへの表示は、たとえば、ホストOS1 0 1 7 自体が備える機能を利用して行えばよい。

【0 0 4 4】

本実施例を「追記」のみを許すように設定し、このテキストエディタプログラムが、ファイルを開くことなく追記する、ことに対応しているときには、ユーザUはファイルF1に追記可能である。しかし、一旦ファイルの内容を読み込んでからでないと、編集(この場合は追記)できないテキストエディタプログラムの場合には、ファイルを読み込む(開く)ことが許可されないため、結果的に追記できない。

【0 0 4 5】

同一ハードディスク内でのファイル移動のように、ハードディスクA 1 0 1 3 内のデータをメモリA 1 0 0 5 内に読み込むことなく、(パス情報を含むファイル名の書き換えを行うだけ、すなわち、ファイルそのものを移動せず、ファイル名とそのファイルがハードディスク上のどこにかかっているか、を示す対応表の部分だけを書き換えることで実現できる場合がある。このようなファイルアクセスについては、任意のユーザが行えるようにしてもよい。ただし、このとき、ファイルリスト1 0 4 3 の整合性を保つために、ファイルリスト1 0 4 3 内のファイル名6 0 0 1 の部分を、当該ファイル移動に合わせて書き換えるようにする。

【0 0 4 6】

また、ファイルのコピーについても、コピーの途中で当該ファイルのデータを不正に読み取られる危険性がない場合、たとえば、ハードディスク自体がファイルコピーの機能を持っているような場合には、任意のユーザが行えるようにしてもよい。ただし、このときファイルのセキュリティレベルも同時にコピーする。すなわち、ファイルリスト1 0 4 3 内に、当該ファイルに関するエントリを追加して、コピー元と同じセキュリティレベルを設定する必要がある。

また、フロッピーディスクなどの取り外し可能な可搬型記憶媒体に移動またはコピーする場合には、ファイル移動(またはコピー)時に、当該ファイルのセキュリティレベルに合わせて暗号化するようにすればよい。さらに、ファイルリスト1 0 4 3 内の当該ファイルに関する情報も、可搬型記憶媒体に書き込むように

してもよい。

これらの処理は、たとえば、アクセス制御プログラム 1 0 3 5 に機能を追加することによって、実現可能である。

【 0 0 4 7 】

図 4 を用いて、本実施例におけるユーザラベル設定時における処理フローを説明する。

(ユーザラベル設定プログラム 1 0 2 3 (ホスト OS 1 0 1 7 上)の動作)

ステップ 4 0 0 1 : はじめ

(ユーザリスト管理プログラム 1 0 3 7 (ゲスト OS 1 0 1 9 上)の処理)

ステップ 4 0 0 2 : ホスト OS 1 0 1 7 上の起動中のプロセスチェックを行う。信頼できないプロセスが起動している場合は、ステップ 4 0 0 3 へ。そうでなければステップ 4 0 0 4 へ

ステップ 4 0 0 3 : おわり(設定失敗)

ステップ 4 0 0 4 : 乱数 R を生成し、ホスト側へ送信する

(ユーザラベル設定プログラム 1 0 2 3 (ホスト OS 1 0 1 7 上)の動作)

ステップ 4 0 0 5 : 設定情報 Cu (ユーザ名、当該ユーザのクリアランスレベルなど) の入力を求める

ステップ 4 0 0 6 : セキュリティ管理者のパスワード P の入力を求める

ステップ 4 0 0 7 : 乱数 R と設定情報 Cu を結合(concatinate)したものを、入力されたパスワードを鍵として暗号化し、ゲスト側へ送信する

(ユーザリスト管理プログラム 1 0 3 7 (ゲスト OS 1 0 1 9 上)の処理)

ステップ 4 0 0 8 : あらかじめゲスト OS 1 0 1 9 側で管理・保持されていたセキュリティ管理者のパスワードを用い、ゲスト側から送られてきた暗号文を復号する。復号した情報に含まれている乱数が、ステップ 4 0 0 4 で生成した乱数 R と等しくなければ、ステップ 4 0 0 9 へ。等しければ、ステップ 4 0 1 0 へ

ステップ 4 0 0 9 : おわり(設定失敗)

ステップ 4 0 1 0 : ユーザリスト 1 0 4 1 にステップ 4 0 0 8 で復号された設定情報を反映させる

ステップ 4 0 1 1 : おわり(設定成功)。

【 0 0 4 8 】

図 5 を用いて、本実施例におけるファイルラベル設定時における処理フローを説明する。

(ファイルラベル設定プログラム 1 0 2 5 (ホスト OS 1 0 1 7 上)の動作)

ステップ 5 0 0 1 : はじめ

(ファイルリスト管理プログラム 1 0 3 9 (ゲスト OS 1 0 1 9 上)の処理)

ステップ 5 0 0 2 : ホスト OS 1 0 1 7 上の起動中のプロセスチェックを行う。信頼できないプロセスが起動している場合は、ステップ 5 0 0 3 へ。そうでなければステップ 5 0 0 4 へ

ステップ 5 0 0 3 : おわり(設定失敗)

ステップ 5 0 0 4 : 乱数 R を生成し、ホスト側へ送信する

(ファイルラベル設定プログラム 1 0 2 5 (ホスト OS 1 0 1 7 上)の動作)

ステップ 5 0 0 5 : 設定情報 Cf (ファイル名、当該ファイルのセキュリティレベルなど) の入力を求める

ステップ 5 0 0 6 : セキュリティ管理者のパスワード P の入力を求める

ステップ 5 0 0 7 : 乱数 R と設定情報 Cf を結合(concatinate)したものを、入力されたパスワードを鍵として暗号化し、ゲスト側へ送信する

(ファイルリスト管理プログラム 1 0 3 9 (ゲスト OS 1 0 1 9 上)の処理)

ステップ 5 0 0 8 : あらかじめゲスト OS 1 0 1 9 側で管理・保持されていたセキュリティ管理者のパスワードを用い、ゲスト側から送られてきた暗号文を復号する。復号した情報に含まれている乱数が、ステップ 5 0 0 4 で生成した乱数 R と等しくなければ、ステップ 5 0 0 9 へ。等しければ、ステップ 5 0 1 0 へ

ステップ 5 0 0 9 : おわり(設定失敗)

ステップ 5 0 1 0 : ファイルリスト 1 0 4 3 にステップ 5 0 0 8 で復号された設定情報を反映させる

ステップ 5 0 1 1 : おわり(設定成功)。

【 0 0 4 9 】

本実施例においては、ユーザラベルまたはファイルのセキュリティレベルの設定を行うときに、ゲスト OS 1 0 1 9 側のユーザリスト管理プログラム 1 0 3 7 ま

たはファイルリスト管理プログラム1039とセキュリティ管理者との間の認証を、チャレンジレスポンス認証(challenge-response identification)と呼ばれる方法を用いておこなっている。さらに、認証と設定情報の受け渡しを同時に行うことにより、認証処理をバイパスする等の不正を防ぎ、かつ、メッセージのやりとりの回数を減らすことができる。

【0050】

チャレンジレスポンス認証については、

文献[HAC]: Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, “Handbook of Applied Cryptography”, CRC Press

に詳しい。なお、チャレンジレスポンス認証で利用する、暗号化処理は、たとえばDES(The Data Encryption Standard)

をCBCモード(Cipher-block chaining mode)で利用することによって行えばよい(文献[HAC]参照)。DES以外の共通鍵暗号を用いてもよい。本実施例では、上記文献[HAC]に開示されたチャレンジレスポンス認証以外の認証方法(たとえば公開鍵暗号に基づく方法)を利用することも可能である。

【0051】

以上のように、本実施例は、ユーザが直接利用する一般的なOSであるホストOS1017とは異なるゲストOS1019上に、強制アクセス制御機能を設けることにより、ホストOS1017上のアプリケーションプログラム1021に改変を加えることなく強制アクセス制御機能を実現可能にしつつ、さらに強制アクセス制御機能に対する不正を防止する。

また、ホストOS1017の多機能化・高機能化等の改良が行われた際にも、構成を変えることなく強制アクセス制御を可能とする。

さらに、ユーザのクリアランスやファイルのセキュリティレベルの設定にあたっては、(一般にはホストOS1017側の管理者とは異なる)セキュリティ管理者の認証を行うことにより、ホストOS1017側の管理者による不正の脅威を減らすことができるようになる。

【0052】

(第2の実施例)

本発明の他の実施の形態を、以下、図を用いて説明する。

【 0 0 5 3 】

前述したように、第 1 の実施例に開示したシステムが正常に起動した場合、ファイルへの不正アクセスは非常に困難になり、システムを安全に運用することが可能となる。このシステムに対して考えられる更なる脅威として、たとえば、システム起動以前にホスト OS のプログラムが記述されたファイルを改竄し、第 1 の実施例に示したゲスト OS 側のアクセス制御機能を回避するという不正行為が考えられる。本実施例はこの不正行為を防止する方法を提供する。

【 0 0 5 4 】

図 9 は、本実施例におけるコンピュータシステムの構成例である。図 1 と同じであるが、ホスト OS 1 0 1 7 管理下のハードディスク A 1 0 1 3 内のファイルが暗号化されている点（暗号化されたファイル 9 0 0 1）、および、ゲスト OS 1 0 1 9 管理下に、暗号プログラム 9 0 0 3、復号プログラム 9 0 0 5、および、暗復号化鍵（対称鍵（symmetric key）ともいう）9 0 0 7 が追加されている点が異なる。

【 0 0 5 5 】

本実施例においては、ファイル書き込み時には、まず、第 1 の実施例と同様にしてアクセス制御を行い、さらに、正当なアクセスであった場合には、ゲスト OS 側の暗号プログラム 9 0 0 3 によって暗号化されてから、ホスト OS 1 0 1 7 側のハードディスク A 1 0 1 3 内に保存される。

ファイル読み込み時には、まず、第 1 の実施例と同様にしてアクセス制御を行い、さらに、正当なアクセスであった場合には、ホスト OS 1 0 1 7 側のハードディスク A 1 0 1 3 に保存された当該ファイル（暗号化されたファイル）はゲスト OS 側の復号プログラム 9 0 0 5 によって復号されたのち、ホスト OS 1 0 1 7 側のアプリケーションプログラム 1 0 2 1 に渡される。

【 0 0 5 6 】

この結果、ホスト OS 側のファイルアクセスが、ゲスト OS 側のアクセス制御プログラムを利用するようになっている場合には（すなわち、ホスト OS 側のアクセス制御プログラムをバイパスしていないときには）、ハードディスク A 1 0 1 3 内の

ファイルは、常に暗号化された状態でありながら、アプリケーションプログラム 1 0 2 1 は、正当なアクセスである限り、暗号化されていることを意識することなく、透過的に利用可能となる。

【 0 0 5 7 】

一方、たとえば、ホストOSがメモリにロードされるより前に、ホストOSのプログラムが記述されたファイルを改竄することにより、不正なホストOSを起動し、ゲストOS側のアクセス制御機能を回避するという不正行為を行った場合には、読み出されたファイルは暗号化されているため、實際上意味をなさない(情報の流出はおこらない)。したがって、結果的には、ゲストOS、および、当該OS上のアクセス制御機能の起動が保証される。さらには、これにより、たとえハードディスク装置ごと取り外し、別のコンピュータに接続してファイルを解析しようとしても、ファイルが暗号化されているため解析できないという効果もある。

【 0 0 5 8 】

なお、さらにホストOSの不正がないことを、より確実に確認するために、ゲストOS上に、ホストOSの検証機能を設けてもよい。これは、たとえば、ゲストOSの管理下に、あらかじめ、改竄されていないホストOSプログラムのデジタル署名を保持しておき、ホストOSの起動時に、ゲストOS側において、実際に起動されるホストOSプログラムが改竄されていないことを、当該デジタル署名の検証手続きによって確認する入出力処理監査処理部をゲストOS側に設ければよい。また、入出力処理監査処理部が、ホストOS側のプロセス管理機能を利用し、ホストOS管理で不正なプロセスが実行されていないことを監査してもよい。

【 0 0 5 9 】

もしホストOSプログラムが改竄されている場合には、ゲストOSに指示して、あらゆるファイルアクセスを禁止する、あるいは、ホストOSの実行を止める、などにより、ホストOSの機能を無効化すればよい。このとき、さらにコンピュータに接続されたディスプレイ上への表示、あるいは、コンピュータに接続されたスピーカーを利用した警告音、あるいは、その他の外部装置を利用したメッセージ等により、ホストOSが改竄されているために起動できない旨、ユーザに伝えるようにしてもよい。

【0060】

図10を用いて、本実施例におけるファイルアクセス制御フロー（読み出し）を説明する。

ステップ10001：はじめ

ステップ10002：ステップ2002～2004と同様

ステップ10003：アクセス制御プログラムは、正当なアクセスか否かを判定する（ステップ2005と同様）。正当であればステップ10006へ。そうでなければステップ10004へ

ステップ10004：ステップ2006と同様

ステップ10005：おわり（アクセス不可）

ステップ10006：ステップ2008～2009と同様

ステップ10007：デバイスドライバは、ハードディスクから

暗号化されたファイルの読み出しを行いファイルフックプログラムに送る

ステップ10008：ファイルアクセスフックプログラムは、OS間通信プログラムを利用し、復号プログラムに、暗号化されたファイルを送る

ステップ10009：復号プログラムは、ゲストOS側で管理された暗復号化鍵を用いて、暗号化されたファイルを復号し、OS間通信プログラム、ファイルアクセスフックプログラムを通じて、アプリケーションプログラムに送る

ステップ10010：おわり（アクセス成功）。

【0061】

図11を用いて、本実施例における、ファイルアクセス制御フロー（書き込み）を説明する。

ステップ11001：はじめ

ステップ11002：ステップ2002～2004と同様

ステップ11003：アクセス制御プログラムは、正当なアクセスか否かを判定する（ステップ2005と同様）。正当であればステップ11006へ。そうでなければステップ11004へ

ステップ11004：ステップ2006と同様

ステップ11005：おわり（アクセス不可）

ステップ 1 1 0 0 6 : ゲスト OS 上の暗号プログラムは、ゲスト OS 側で管理された暗復号化鍵を用いて、アプリケーションプログラム 1 0 2 1 が書き込もうとしている情報を暗号化する

ステップ 1 1 0 0 7 : ステップ 2 0 0 8 ~ 2 0 1 0 と同様

ステップ 1 1 0 0 8 : おわり (アクセス成功)

ファイル生成時におけるファイルアクセス制御の処理フローは、図 1 1 に示したファイルアクセス制御フロー (書き込み) と基本的に同様であるが、ステップ 1 1 0 0 2 が、

ステップ 1 1 0 0 2 : ステップ 3 0 0 2 ~ 3 0 0 5 と同様
となる点が異なる。

【 0 0 6 2 】

ステップ 1 0 0 0 9 の復号化処理、および、ステップ 1 1 0 0 6 の暗号化処理では、たとえば DES (The Data Encryption Standard) を CBC モード (Cipher-block chaining mode) で利用すればよい (文献 [HAC] 参照)。ただし、CBC モードを利用する場合であっても、最初のブロック長 (DES の場合であれば 64-bit) 分については、同じ平文を同じ鍵で暗号化すると、同じ暗号文となってしまう。これを避けるためには、CBC モードで利用される初期値 IV をファイルごとに異なるようにすればよい。たとえば、ステップ 1 1 0 0 6 で暗号化をする際に 64-bit の乱数を生成し、それを IV として暗号化時に利用するとともに、この IV をファイルリストに項目を追加して保存する。ステップ 1 0 0 0 9 中の復号化処理では、ファイルリストから復号対象ファイルに対応する IV を読み出し復号時に利用すればよい。

【 0 0 6 3 】

本実施例では、ひとつの暗復号化鍵を用いて、暗復号化を行っていたが、複数の暗復号化鍵を用いてもよい。たとえば、ファイルのセキュリティレベル・区画ごとに異なる鍵、あるいは、異なる暗号方法を用いてもよい。本実施例を、このように変更した場合、セキュリティレベルが 4 段階あり、区画が 4 区画あるので、全部で 1 6 個の異なる暗復号化鍵を用いることになる。これらの鍵はゲスト OS 側で管理すればよい。さらには、ファイルのセキュリティレベルに応じて、暗号方法、あるいは、鍵として、安全性が異なるものを用いてもよい。(たとえば、

セキュリティレベルが高いファイルに対しては、長い鍵長の暗号化鍵を用いてもよい)。

【 0 0 6 4 】

また、本実施例で用いる暗復号化鍵は、ハードディスクB1015とは異なるハードウェアモジュール(例:ICカードなど)に格納しておくようにしてもよい。この場合、さらに当該ハードウェアモジュール内に、ゲストOS1019およびその上で動作する各種プログラム(アクセス制御プログラム1035など)が改変されていないことを検証する機能を持たせておき、正当であると確認されたときに限り、暗復号化鍵が利用可能になるように(たとえば、メモリB1007内にロードされるように)しておく、ゲストOS側の改ざんという脅威にも対抗できるため、より効果的である。

【 0 0 6 5 】

また、本実施例では、ファイルの内容自体を暗号化していたが、そうではなく、ファイルの管理テーブル部を暗号化するようにしてもよい。ファイルの管理テーブルは、通常ハードディスク内の、ファイルとは別領域に記憶されており、ファイル名と、ハードディスク内のどの部分に当該ファイルが保存されているかを示すアドレス情報との対応が、記述されている。したがって、このアドレス情報部分を暗号化することにより、ディスク内のどこにファイル情報が保存されているか調べることが困難になるため、ファイル情報漏洩の防止に効果がある。また、暗号化する対象となるデータサイズが小さいため、暗復号化にかかる処理量が少なくてすむ、という利点がある。

【 0 0 6 6 】

なお、本実施例でも、第1の実施例と同様、ファイルアクセスフックプログラム1029が、I/Oマネージャ1027のファイルシステムドライバ1031に対するファイルアクセス命令を検知し、そのファイルアクセス命令をファイルシステムドライバ1031に渡す前に、ゲストOS1019上のアクセス制御プログラム1035に渡す例を示した。しかし、これとは異なってもよい。たとえば、ファイルアクセスフックプログラム1029は、ファイルアクセス(例えば、読み出し)命令を、ファイルシステムドライバ1031に渡すのと同様、ある

いは後に、アクセス制御プログラム1035に渡すようにし、当該ファイルアクセスが正当なものでなかったときは、ステップ10009の復号化処理を行わないようにしてもよい。このようにしても、不当な読み出し処理をしようとする、アプリケーションプログラムにとっては、暗号化されたままのファイルが読み込まれることになるため、情報の漏洩はおこらない。

【0067】

ゲストOSに対しては、上記第1、2の実施例による強制アクセス制御機能だけでなく、さらに別の機能を追加してもよい。

たとえば、ホストOS側でどのようなファイルアクセスがあったか（アクセス時刻、アクセスしようとしたユーザ名・プログラム名、アクセス対象ファイル名、アクセスの種類、ファイルになされた変更内容など）を記録し、保持する機能（ファイルアクセスログ管理機能）をゲストOS側が備えてもよい。

ゲストOS側で実現することにより、ホストOS、および、ホストOS上で動作するアプリケーションプログラムを改変することなく、ファイルアクセスを記録することができるようになり、さらに、記録されたファイルアクセスログが、ホストOS側から改竄、または、消去される危険性がなくなる。

【0068】

また、ホストOS上のコンピュータウィルスを検知、あるいは、駆除するコンピュータウィルス対策プログラムをゲストOS側に設けてもよい。さらには、検知したウィルスに関する情報をネットワークを通じて、他のコンピュータとの間で送受信する機能をゲストOSに設けてもよい。

【0069】

あるいはまた、ホストOS側のアプリケーションプログラムから明示的に実行が要求される機能を実現してもよい。たとえば、デジタル署名生成機能または公開鍵暗号の復号機能などの暗号機能、および、これらの機能において利用される秘密鍵情報の管理などが挙げられる。さらには、発明者等が特願2000-313123号に開示した、デジタル署名生成時に、その時点での署名履歴を利用する署名方法、および、署名履歴管理方法を実現してもよい。

この場合は、上述のファイルアクセス制御機能や、ファイルアクセスログ管理

機能などとは異なり、ホストOS側アプリケーションプログラムが、あらかじめ、このゲストOS側で実現された機能に対応している必要がある。ただし、実行にあたっては、上記のユーザラベル設定フローやファイルラベル設定フローと同様の方法によって、固有のユーザ認証を行うことにより、ホストOS側のシステム管理者による不正利用を防ぐことができる。したがって、前述のデジタル署名生成機能や、公開鍵暗号の復号機能のような、各ユーザの秘密鍵情報を利用する機能を実現するためには、特に適している。あるいは、上記のコンピュータウィルス対策プログラムをホストOS側のアプリケーションプログラムから明示的に実行が要求される機能として実現してもよい。

【0070】

上記各実施例においては、複数OS制御技術を利用し、ユーザが直接利用するホストOS上のファイルアクセス制御を、ホストOSとは異なるゲストOS管理下のソフトウェアが実現していた。しかし、本発明はこれに限定されない。たとえば、ゲストOSとゲストOS上に設けられた機能（処理部）のいずれか一つ以上を、ハードウェアとして実現してもよい。この場合、さらに、このハードウェア内に、ユーザリスト、ファイルリスト、セキュリティ管理者のパスワード、ファイル暗号用の暗号化鍵等を記憶するようにしてもよい。

【0071】

【発明の効果】本発明によれば、情報処理システムの情報セキュリティに関する安全性を、利用者の利便性を損なうことなく、向上させることが可能となる。

【図面の簡単な説明】

【図1】本発明の第1の実施例を実現する計算機の概略構成図である。

【図2】第1の実施例におけるファイルアクセス（読み出し、書き込み）制御の処理フロー図である。

【図3】第1の実施例におけるファイルアクセス（ファイル作成）制御の処理フロー図である。

【図4】第1の実施例におけるファイルアクセス（読み出し、書き込み）制御の処理フロー図である。

【図5】第1の実施例におけるファイルラベル設定時における処理フロー図であ

る。

【図6】第1の実施例におけるユーザリストユーザリストの一例である。

【図7】第1の実施例におけるファイルリストファイルリストの一例である。

【図8】第1の実施例におけるユーザ”researcher”がアクセスすることが許されるファイルのセキュリティレベルを示したものである。

【図9】第2の実施例を実現する計算機の概略構成図である。

【図10】第2の実施例におけるファイルアクセス(読み出し)制御の処理フロー図である。

【図11】第2の実施例におけるファイルアクセス(書き込み)制御の処理フロー図である。

【符号の説明】

1001: コンピュータ

1003: CPU

1005: メモリA

1007: メモリB

1009: ディスプレイ

1011: キーボード

1013: ハードディスクA

1015: ハードディスクB

1017: ホストOS

1019: ゲストOS

1021: アプリケーションプログラム

1023: ユーザラベル設定プログラム

1025: ファイルラベル設定プログラム

1027: I/Oマネージャ

1029: ファイルアクセスフックプログラム

1031: ファイルシステムドライバ

1033: デバイスドライバ

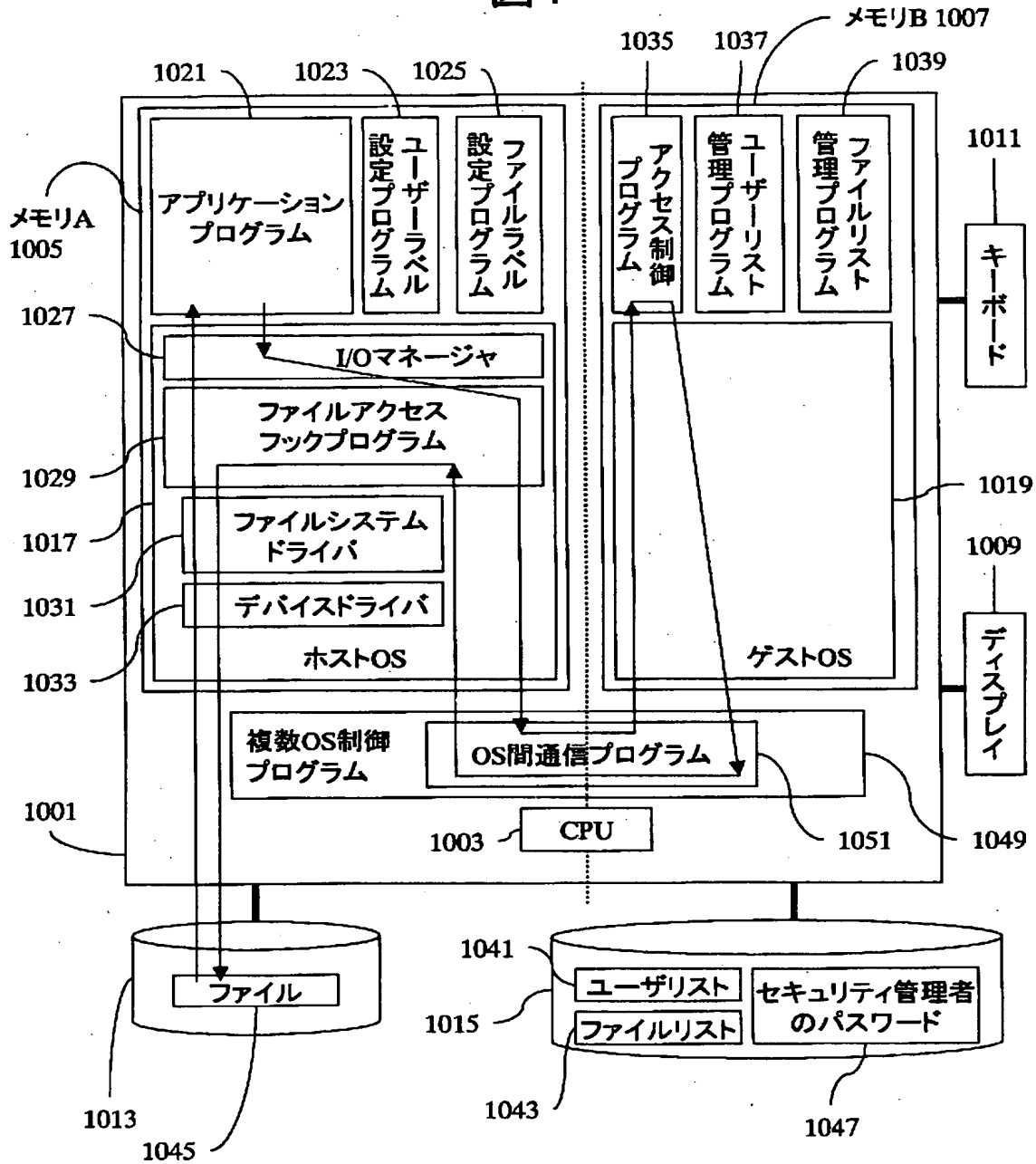
1035: アクセス制御プログラム

1037 : ユーザリスト管理プログラム
1039 : ファイルリスト管理プログラム
1041 : ユーザリスト
1043 : ファイルリスト
1045 : ファイル
1047 : セキュリティ管理者のパスワード
1049 : 複数OS制御プログラム
1051 : OS間通信プログラム
9001 : 暗号化されたファイル
9003 : 暗号プログラム
9005 : 復号プログラム
9007 : 暗号化鍵

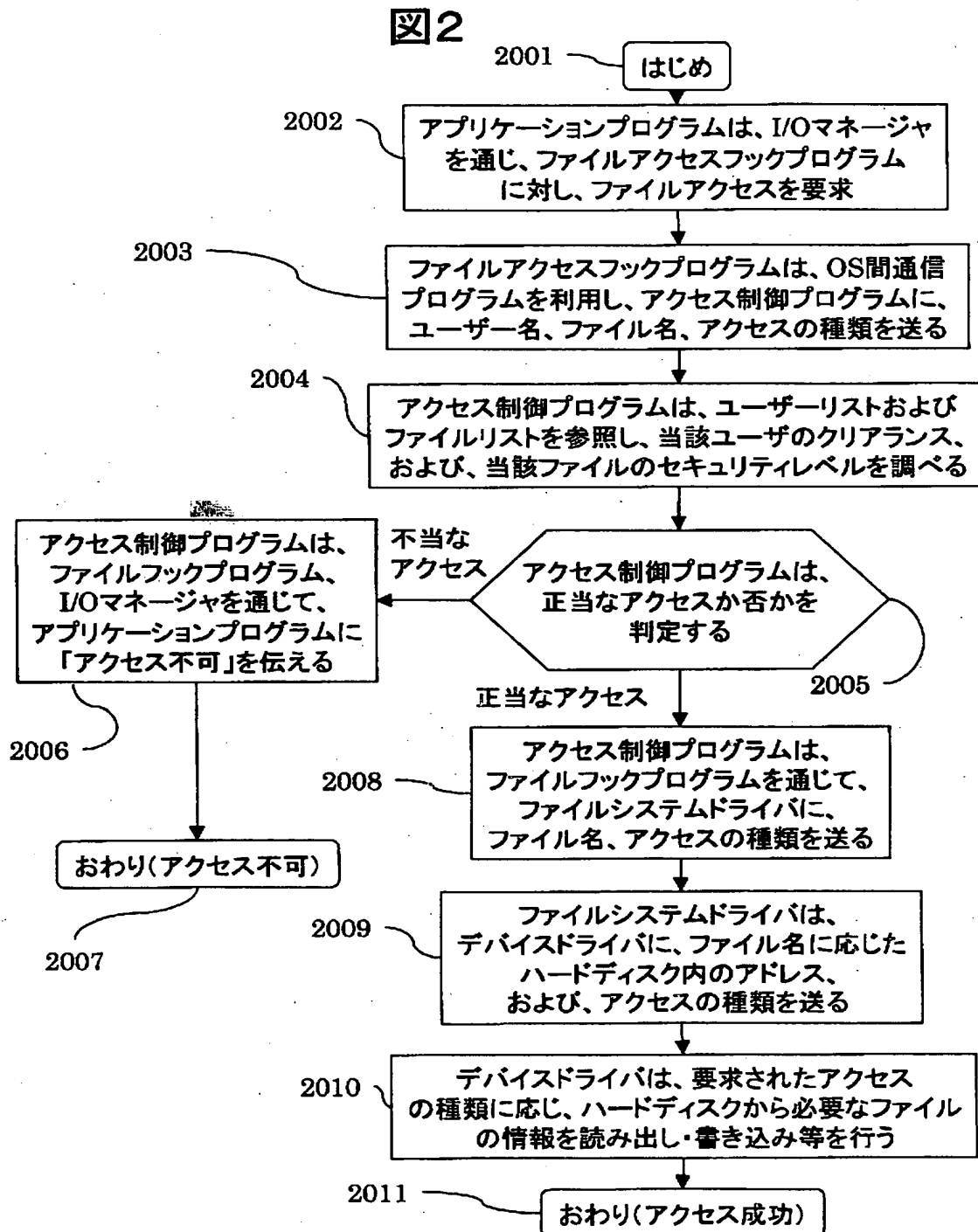
【書類名】 図面

【図 1】

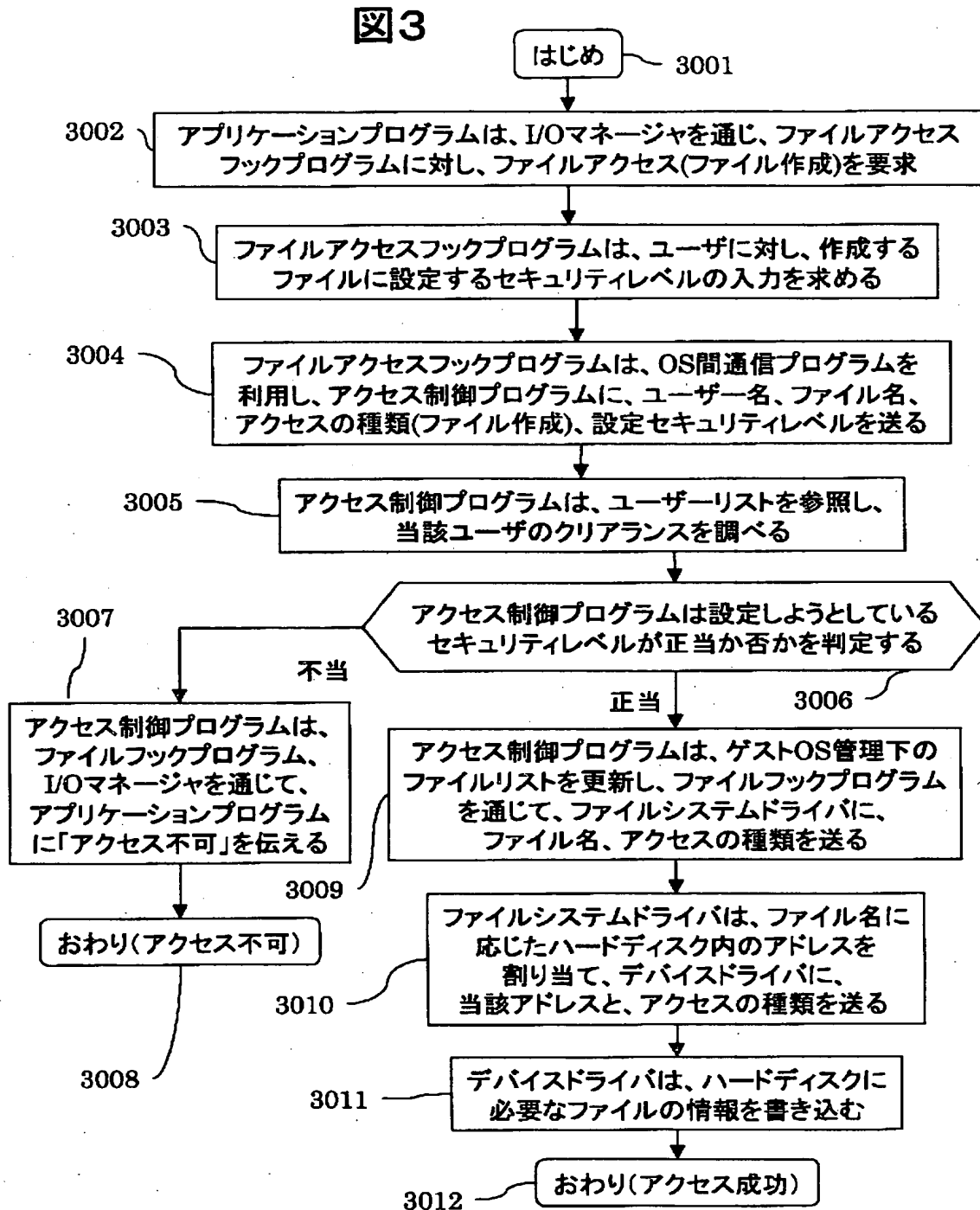
図 1



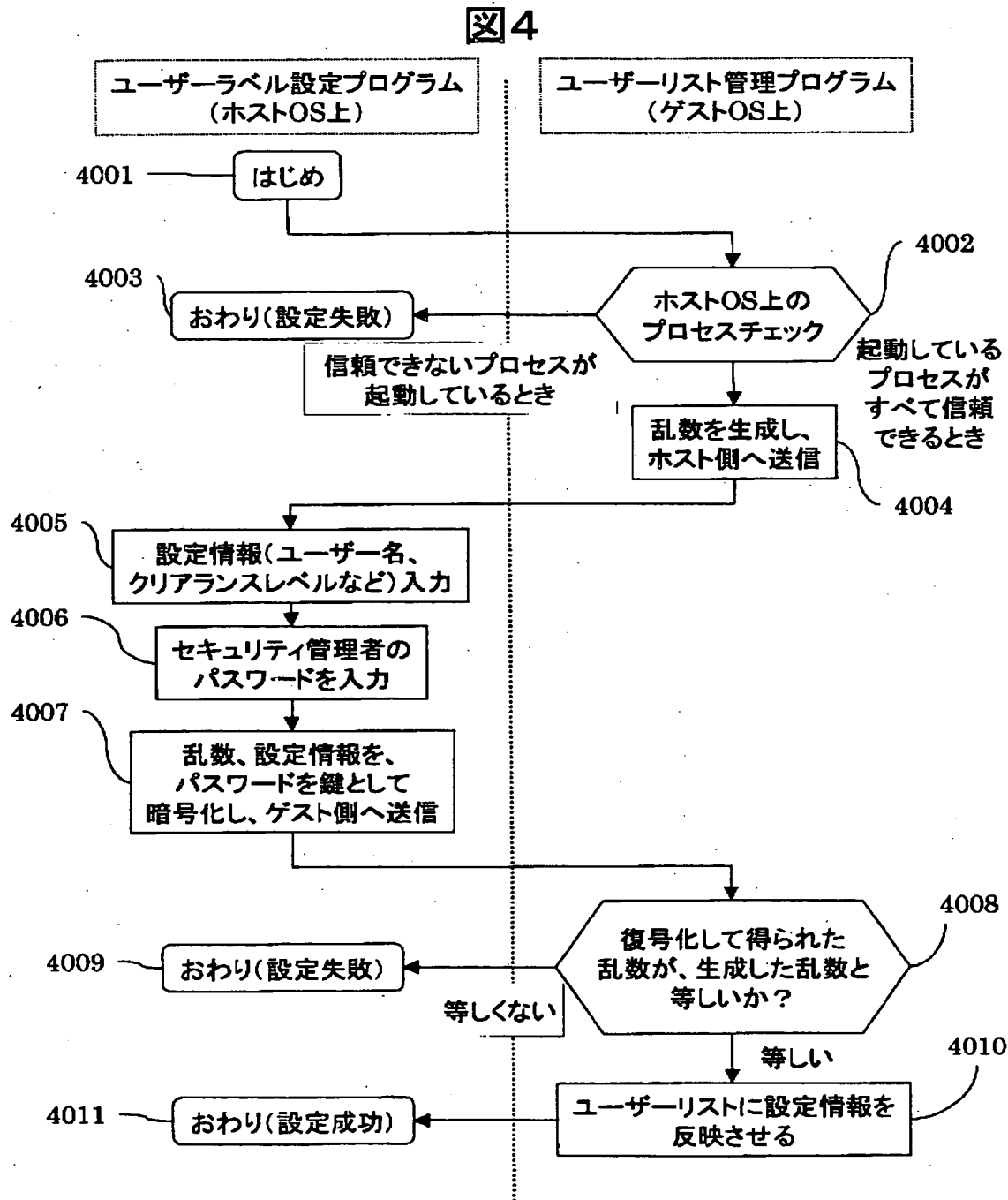
【図 2】



【図3】

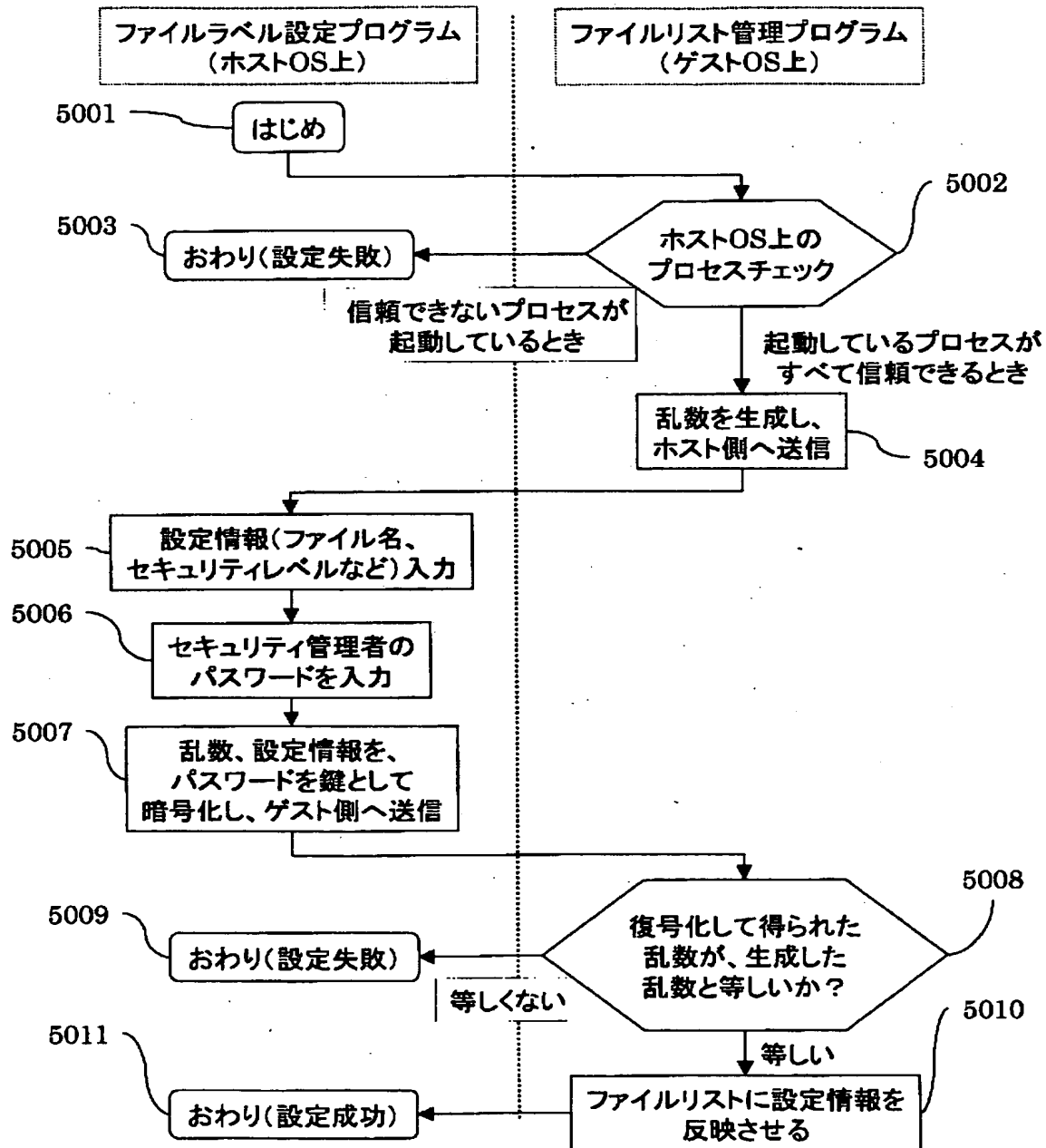


【図4】



【図 5】

図 5



【図 6】

図 6

ファイル名	セキュリティレベル
C:\home\user1\secret.txt	営業部門・極秘、企画部門・極秘
C:\home\user2\research_report.doc	研究部門・極秘
D:\public_data\public.bmp	人事部門・一般、営業部門・一般、 企画部門・一般、研究部門・一般
D:\secret_data\top_secret.doc	人事部門・機密
...	...

【図 7】

図 7

ユーザ名	クリアランス
researcher	研究部門・極秘、企画部門・秘密
guest	人事部門・一般、営業部門・一般、 企画部門・一般、研究部門・一般
president	人事部門・機密、営業部門・機密、 企画部門・機密、研究部門・極秘
personnel_director	人事部門・機密
...	...

【図 8】

図 8

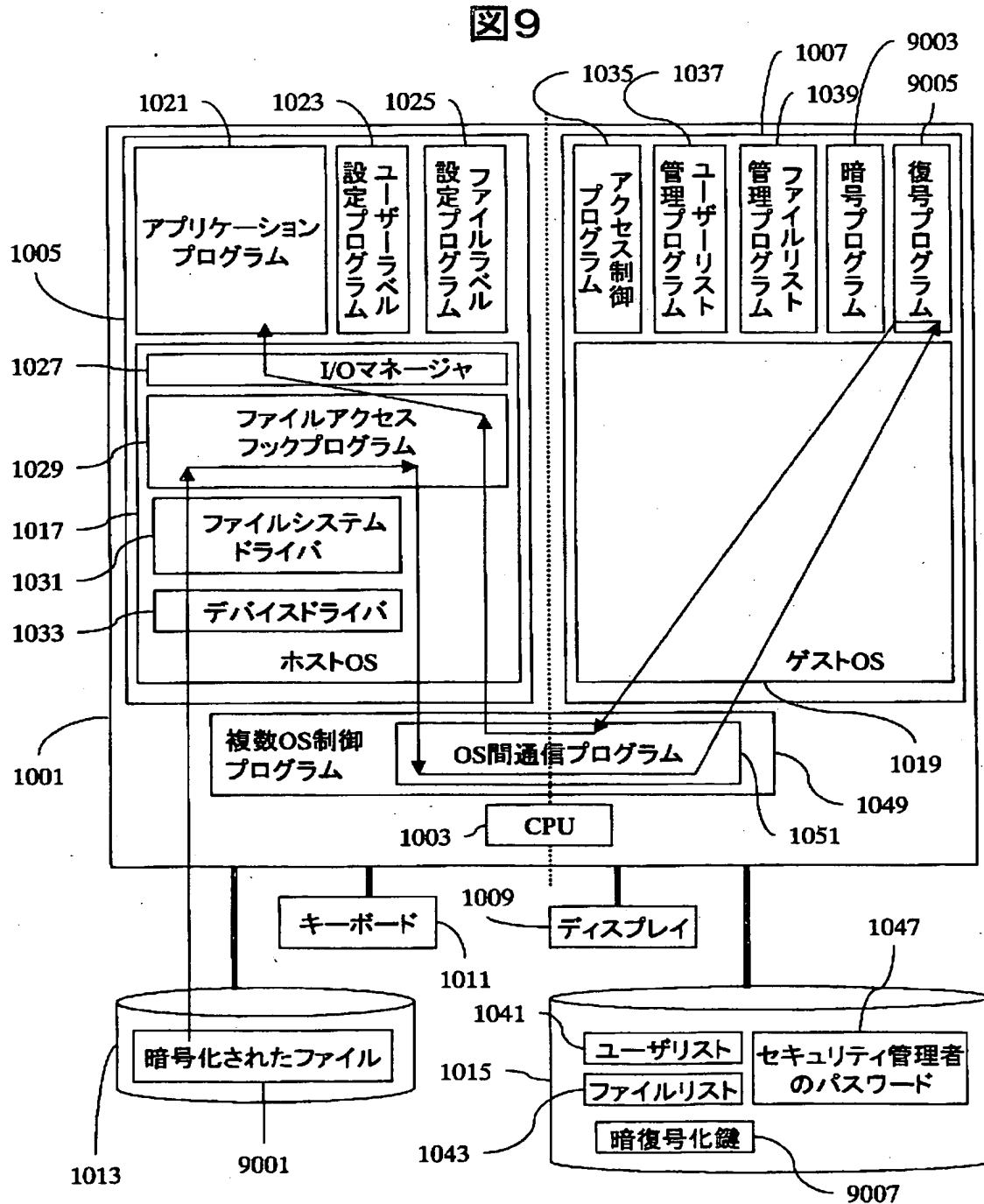
“researcher”が読み取り可能な
ファイルのセキュリティレベル

		8001 人事部門	8003 営業部門	8005 企画部門	8007 研究部門
8009					
8011	機密				
8013	極秘				○
8015	秘密			○	○
	一般			○	○

“researcher”が書き込み、または、
作成可能なファイルのセキュリティレベル

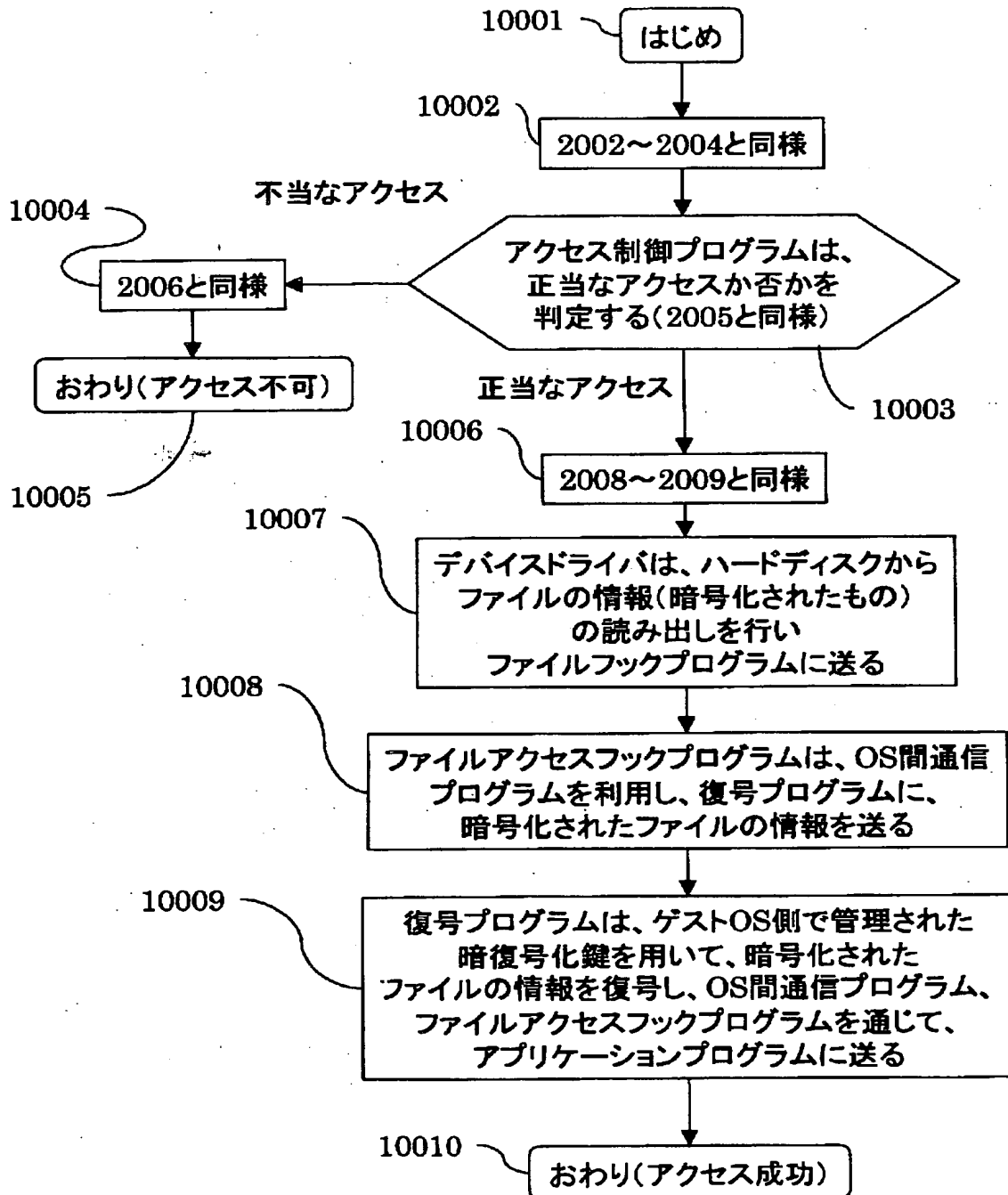
		8021 人事部門	8023 営業部門	8025 企画部門	8027 研究部門
8029					
8031	機密			○	○
8033	極秘			○	○
8035	秘密			○	
	一般				

【図9】



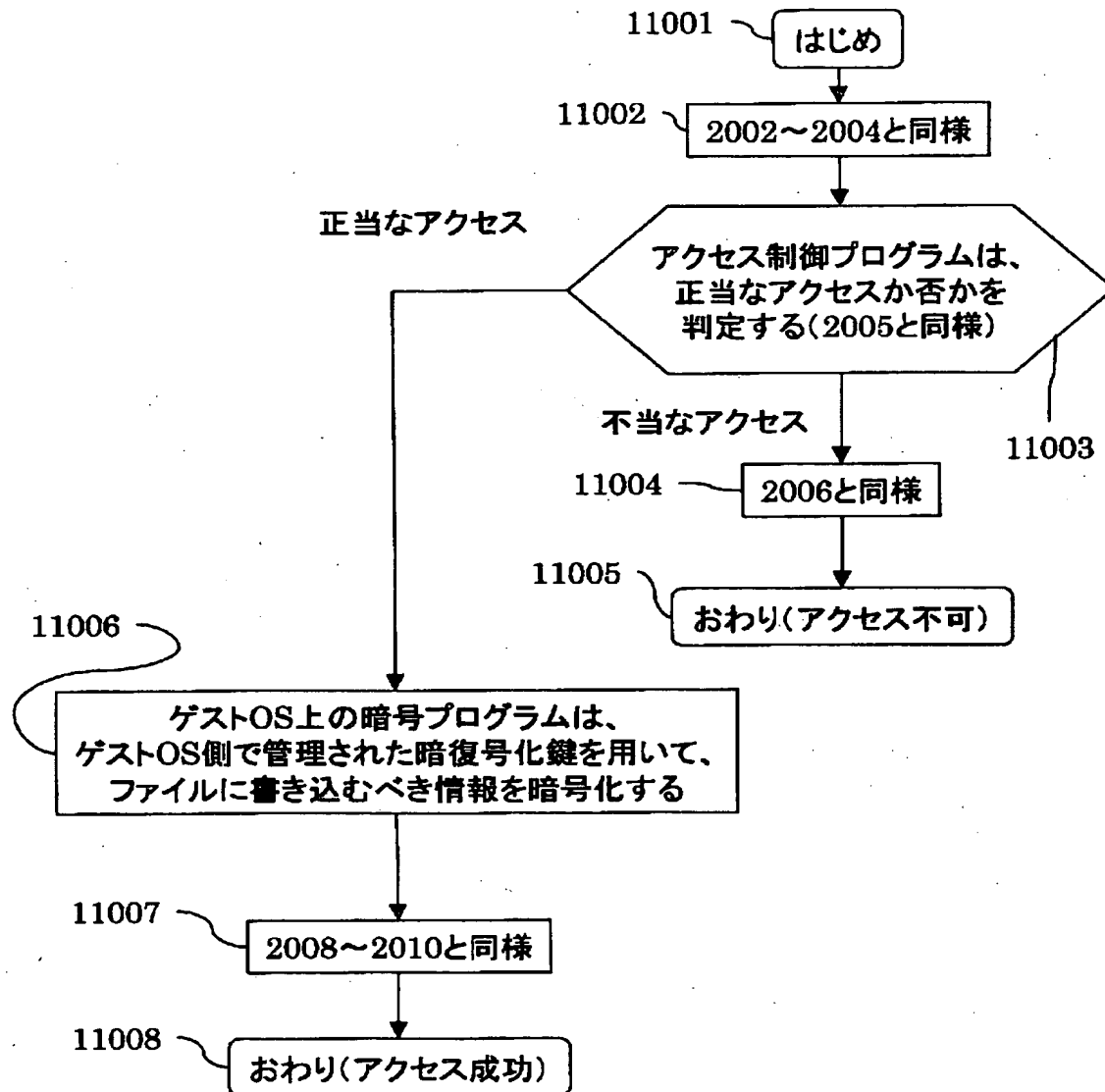
【図10】

図10



【図11】

図11



【書類名】 要約書

【要約】

【課題】 ファイルへの不正アクセスを防ぐ仕組みを任意アクセス制御に基づくアクセス制御機能を備えたOSに依存している情報処理システムに対し、よりコストがかからず、安全性を高められる技術を提供する。

【解決手段】 情報処理システム内にシステム管理者とは異なる特定のユーザだけが利用できる機能・領域を設けるための方法、および、装置を提供し、さらに、当該領域にアクセス制御機能を設けることにより、アクセス制御機能に対する不正を防止。

【選択図】 図 1

認定・付加情報

特許出願の番号	特願2001-125224
受付番号	50100596125
書類名	特許願
担当官	第七担当上席 0096
作成日	平成13年 4月25日

<認定情報・付加情報>

【提出日】	平成13年 4月24日
-------	-------------

出 願 人 履 歴 情 報

識別番号

[000005108]

1. 変更年月日 1990年 8月31日

[変更理由] 新規登録

住 所 東京都千代田区神田駿河台4丁目6番地
氏 名 株式会社日立製作所